IBM® DB2® Spatial Extender - Spatial data within the RDBMS

David W. Adler

IBM Corporation
2455 South Rd
Poughkeepsie, NY 12601
USA
dadler@us.ibm.com

Abstract

Much of the data that we encounter has a spatial (geographic locational) aspect yet this has not been readily exploited by traditional RDBMS. Over the past five years there has been a confluence of Geographic Information System (GIS) technology, RDBMS architecture and SQL standards that has fostered the implementation of spatial processing within the RDBMS.

This paper will present a brief overview of spatial processing and the evolution of technology leading to the development of the IBM DB2 Spatial Extender that exploits the IBM DB2 Universal Database (UDB) object-relational support to implement a standards-based SQL spatial capability.

1. Introduction

It is said that 80% of the data stored in computers has a spatial aspect [1]. This typically is associated with the location of a customer, office, property or even a mobile device. Most often the location is defined by an address consisting of a house number, street name, city, country and postal code which is not a representation conducive to spatial processing other than simple aggregation by postal code or city.

Spatial processing is nothing new – it dates back over two thousand years to the creation of the first maps and traders planning their routes. Aside from advances in accuracy and the mathematics of cartography, this has remained a manual and visual process until the age of computing.

We would like to be able to pose business queries like: "Which customers are within the territory of my most profitable store?" or "What is the average revenue per square mile of sales territory?". One might imagine possible manual solutions using paper maps but they quickly become impractical for useful data volumes.

2. Evolution of Spatial Processing

The origin of computer-based spatial processing goes back to the 1970s with the development of Automated Mapping and Facilities Management (AM/FM) systems developed primarily for utilities to keep track of real-world features such as pipes, transmission lines and transformers. At the same time, GIS systems were developed to analyze and maintain environmental and cadastral features such as rivers and parcels. Common to these systems is the *geometry* of each of these features which is generally a *point*, *line* or *polygon* composed of one or more geographic location (coordinate) values.

Initially these AM/FM/GIS or more simply, GIS systems were implemented on standalone workstations using proprietary interfaces and data representations on simple file systems. This is shown in figure 1 as *First Generation*. Although the spatial functionality may be sufficient for application needs, it is not an open system amenable to enterprise solutions.

In the early 1990s with the acceptance of RDBMS technology and greater need for enterprise data access/sharing, major GIS vendors such as IBM [2], ESRI® [3] and MapInfo® [4] developed middleware products that used an RDBMS as the spatial data repository. These products provided to applications the RDBMS benefits of concurrency, data backup/recovery and client/server capability. All the intelligence for spatial indexing and geometry representation was

**Proceedings of the 27th VLDB Conference,
Roma, Italy, 2001**

embedded in the middleware that was constrained to use basic SQL operations and generic datatypes such as Integer or BLOB. A disadvantage is that applications are still constrained to a proprietary GIS API. This is referred to as *Second Generation*.

The customer demand for interoperable solutions not dependent on proprietary APIs and for business applications that could exploit the full RDBMS capability has lead to the development of *Third Generation* systems. These systems move primary data management into the database and use expanded standard SQL to operate on the data. This is discussed later in the context of *SQL and Spatial Standards*.

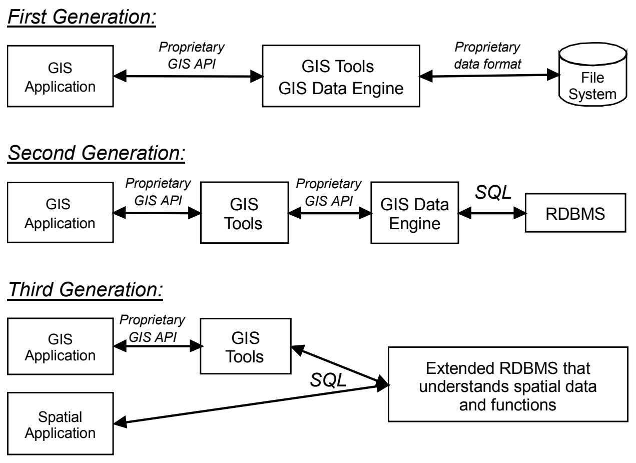


Fig 1: Evolution of GIS environments

3. Spatial Data

The creation of spatial data has been a very high-cost process involving largely manual data conversion from paper maps or survey information, especially when high precision (meter or sub-meter) is required as is the case for utility and cadastral applications. The cost inhibited many business applications that didn't require this degree of precision but had no other source of data.

The breakthrough in the cost and availability of spatial data for business applications came through the involvement of the public sector, the availability of low-cost, reasonable accuracy GPS devices and the development of geocoding technology.

In order to meet mandated governmental tasks such as census taking and map production, national governments have born the cost of developing spatial representations of census blocks, political districts, cities, states, highways, and street addresses. United States Census Bureau has made much of this available at nominal cost through the TIGER [5] data. The private sector has added value by offering TIGER data derivatives in common spatial data formats and enhanced data validation. Other national spatial data infrastructures and public/private sector cooperation are making spatial data increasingly available on a worldwide basis. Current GPS devices allow businesses to collect additional spatial data at very low cost with approximately 10-meter precision.

Geocoding is the process of algorithmically converting a street address to a spatial location by referencing spatial street data that has associated address range information for each street segment. Numerous vendors have provided geocoders for countries that have consistent street addressing schemes. The value of geocoders include:

1. Exploitation of existing databases containing addresses as spatial data sources.
2. Correction of inconsistently represented addresses ('street' vs 'st' vs 'str').
3. Reduction of storage required by only storing street segments and not every known address.

4. SQL and Spatial Standards

Each GIS vendor has defined it's own proprietary representation for spatial data. National governmental organizations have defined 'standard' representations for spatial data that for the most part have not been widely accepted. The result has been that application developers have had great difficulty integrating data from different sources and could not provide applications independent of specific GIS vendor interfaces.

The Open GIS Consortium (OGC) was founded in 1994 with the mission "to deliver spatial interface specifications that are openly available for global use." The OGC is composed of approximately 200 members from industries, governmental agencies and academia who work together in a consensus process. For our consideration here, the "OpenGIS® Simple Features Specification for SQL" [6] is of critical importance. This specifies an object model for the implementation of spatial datatypes and operations in the SQL RDBMS environment. The OGC also operates a specification conformance process that has been used to certify the implementation of this specification by leading GIS tool and database vendors.

The ISO (International Organization for Standardization) is the authoritative body producing SQL standards implemented by RDBMS vendors. There are two standards of particular importance to spatial processing:

1. *SQL99* [7] that provides a mechanism for the creation of user-defined types in the SQL environment. This is a prerequisite for the definition of a spatial object model.
2. *SQL/MM Spatial* [8] which is closely based on the "OpenGIS® Simple Features Specification for SQL", providing the same functionality but as a recognized international standard.

5. DB2 Spatial Extender

The DB2 Spatial Extender [9] brings together a number of key capabilities for supporting spatial processing within the domain of the RDBMS:

1. Standards-based geometry model using object-relational technology.
2. Standards-based geometry functionality
3. Spatial indexing
4. Integration of geocoding
5. Import / export of common spatial data formats.
6. Integration of spatial administration and documentation with DB2 UDB.

Each of these capabilities will be addressed in more detail below.

5.1 Standards-based geometry model

The DB2 UDB object-relational capability allows a developer to define new user-defined structured types (UDT) composed of existing relational types, sub-types that inherit and extend user-defined types and to provide user-defined functions (UDF) at the appropriate levels in the type hierarchy. The DB2 Spatial Extender builds on this to define spatial datatypes and functions that conform to the ISO spatial standard.

One of the first steps is to create the user-defined types corresponding to each of the geometry types defined in the ISO spatial standard. It also creates all of the functions associated with the geometry types. This is all performed by invoking a stored procedure to spatially-enable the database.

Figure 2 shows the type hierarchy that is created. The shaded boxes indicate types that are not instantiable, similar to an abstract class in object-oriented terminology. Non-instantiable types can be specified as database column types and can have functionality associated with them. For example, a column of type ST_Geometry can contain values of ST_Point, ST_LineString, ST_Polygon, etc. The unshaded boxes indicate types that can be directly instantiated.

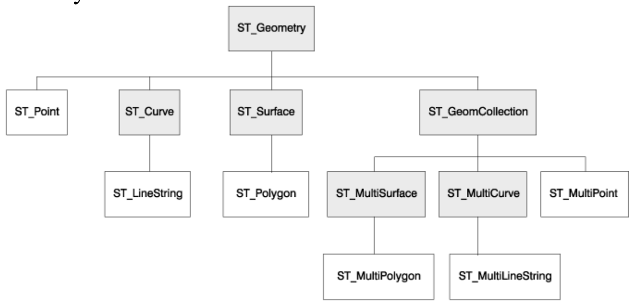


Fig 2 – Spatial datatypes

With this capability, we could define tables such as:

```
CREATE TABLE STORES (
          SNAME CHAR(20),
          REVENUE DOUBLE,
          TERRITORY  ST_POLYGON)
    and
CREATE TABLE CUSTOMERS  (
          CNAME CHAR(20),
          LOCATION ST_POINT)
```

5.2 Standards-based geometry functionality

The ST_Geometry type has associated with it 20 functions such as ST_AsText, ST_Distance, ST_Intersects, etc which are applicable to all sub-types. Each of the subtypes has additional type specific functionality. For example, ST_Polygon has functions such as ST_Area and ST_Centroid that are specific to this type.

If we wanted to get a list of customers within the territory of each store ordered by store revenue, we could write an SQL query like:

```
SELECT SNAME, CNAME
    FROM CUSTOMERS, STORES
    WHERE
          ST_Intersects(TERRITORY,LOCATION)=1
    ORDER BY REVENUE
```

Constructor functions are provided to easily create geometry values from standard datatypes. For example:

1. ST_Point(-73.5, 42.3)
    or
2. ST_LineFromText('linestring(10 10, 20 20)')

The first example creates an ST_Point value from two numeric values representing the longitude and latitude.

The second example creates an ST_LineString value using what is defined in the OGC and ISO standards as the "well-known text" representation for each of the geometry types, a character string containing coordinate pairs.

Similarly, any geometry can be returned to the application as a character string in "well-known text" by using the method ST_AsText.

5.3 Spatial indexing

The multi-dimensional nature of spatial data is not amenable to efficient indexing by directly using the B-tree commonly used for alphanumeric data in RDBMS.

There are many indexing approaches for spatial data, the most common of which include "grid" (used by DB2 Spatial Extender), "quadtree" and "R-tree" which attempt to cluster spatial data by proximity.

Although the Second Generation and Third Generation GIS environments both use the same types of spatial indexing, by implementing the spatial index within the RDBMS, it is possible to incorporate this index knowledge into the query optimization, as implemented in DB2 UDB V7. This can result in significantly better query performance.

5.4 Spatial data import / export

There is a large body of existing spatial data that is available via the Internet or on physical media in *shapefile* format, a de-facto spatial data standard defined by Environmental Systems Research Institute (ESRI).

The DB2 Spatial Extender provides a utility that can import data in shapefile format, creating the appropriate table and populating both alphanumeric and spatial columns.

Similarly, any table containing a spatial column can be exported to shapefile format for use with other applications.

5.5 Geocoding

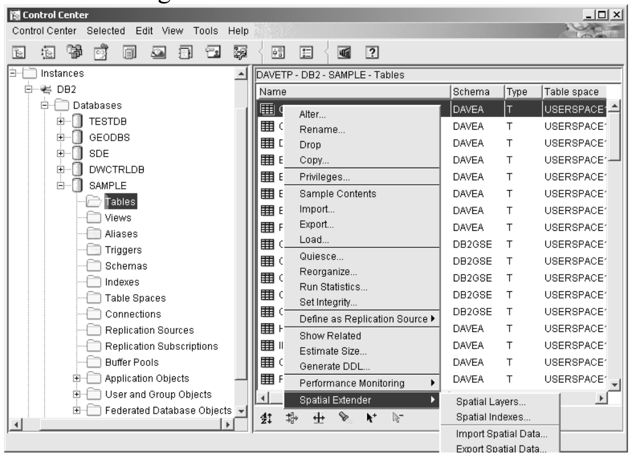
As described above, geocoding is the process of converting an address to a geographic location. When a spatial column is defined, the developer can associate a geocoder that will take as input other columns containing the street address. The geocoder can either be run in batch mode, processing all rows in the table, or in automatic mode, when triggers defined on the address columns detect that a value has changed.

A default geocoder for the United States is provided but geocoders for other countries and with other characteristics can be registered using an open interface.

5.6 Integration with DB2 UDB

DB2 UDB provides a graphical user interface ("Control Center") for administering databases. One can create databases and tables, perform backup/recovery, etc. When the DB2 Spatial Extender is installed, spatial-specific functionality is integrated smoothly with the Control Center to perform operations like spatially-enabling a database, specifying a geocoder or adding a spatial column to a table. An example is shown in Fig 3.

The DB2 Spatial Extender documentation is installed and made available through the DB2 Information Center, a common interface for accessing task, reference, troubleshooting and other information.



DAVETP - DB2 - SAMPLE - Tables

Name	Schema	Type	Table space
C	DAVEA	T	USERSPACE1
C	DAVEA	T	USERSPACE1
D	DAVEA	T	USERSPACE1
E	DAVEA	T	USERSPACE1
E	DAVEA	T	USERSPACE1
E	DAVEA	T	USERSPACE1
E	DAVEA	T	USERSPACE1
F	DAVEA	T	USERSPACE1
C	DB2GSE	T	USERSPACE1
C	DB2GSE	T	USERSPACE1
C	DB2GSE	T	USERSPACE1
C	DB2GSE	T	USERSPACE1
C	DB2GSE	T	USERSPACE1
H	DAVEA	T	USERSPACE1
II	DAVEA	T	USERSPACE1
C	DAVEA	T	USERSPACE1
F	DAVEA	T	USERSPACE1

Fig. 3 – DB2 UDB Control Center

6. Conclusion

A large segment of the VLDB community is focused on techniques for improving the function and performance of Data Warehousing, Data Mining and OLAP applications implemented with an RDBMS.

By providing a native spatial data capability within the RDBMS, a new dimension is added which can be exploited for each of these application areas, extending the business intelligence possibilities for VLDBs.

Emerging technologies such as wireless location-based services can generate huge volumes of spatial data. Assuming that the privacy issues are addressed, one can imagine applications that mine a log of all cellular phone or PDA transactions to develop marketing campaigns based on call location. This type of application would now be possible with DB2 Spatial Extender.

7. References

[1] Daratech. Geographic Information Systems Markets and Opportunities. Daratech, Inc., 2000.

[2] IBM. **geoManager® Relational Database System General Information**. June 1990.

[3] ESRI Spatial Database Engine. http://www.esri.com/software/sde/index.html

[4] Mina Chebel. **MapInfo SpatialWare® a Spatial Information Server for RDBMS**, Proceedings of the 24th VLDB Conference, New York, USA, 1998. http://www.vldb.org/dblp/db/conf/vldb/Mina98.html

[5] U.S. Census Bureau. http://tiger.census.gov

[6] OpenGIS Consortium, Inc. **OpenGIS® Simple Features Specification for SQL, Revision 1.1**. OpenGIS Project Document 99-049. May 1999.

[7] **ISO Final Draft International Standard, Database Language SQL – Part 2: Foundation**, ISO/IEC FDIS 9075-2:1999, March 1999.

[8] **ISO International Standard, Information technology - Database languages - SQL Multimedia and application packages - Part 3: Spatial**, ISO/IEC 13249-3:1999, December 1999.

[9] IBM DB2 Spatial Extender. http://www.software.ibm.com/data/spatial.